Natural Language and Gesture Control for Robot Navigation

Ronald Baker, Zhiyuan (Paul) Zhou

May 2020

Abstract

Using Natural Language and gestures to provide information and give commands is a natural operation performed by human beings. As robots begin to play an increasingly important part in task performing, it is crucial that they understand Natural Language commands along with human gestures to perform tasks more accurately. Current approaches focus solely on processing verbal commands and ignores how gestures can provide extra information to the command. This project aims to create a pipeline that enables a robot to navigate to a destination more accurately, with pointing gestures corroborating verbal language command. Our model will compute the n-most-likely destination that matches with the inputted natural language command, and then use an input pointing-gesture to pick out the most likely one. The pipeline will operate in Unity, where the natural language command is given by the user's verbal language, and the gesture command can be inputted by pointing gesture in Virtual Reality using the Vive headset or simulating a pointing action by clicking the mouse in a certain direction. With our approach, the robot can navigate to the destination with a higher accuracy and can distinguish between vague language commands.

Introduction

As robots play a more and more important role in our lives, Human-Robot Interaction (HRI) becomes crucial. It is important that robots understand the task given by humans correctly and can perform them accurately. To enable untrained users to work to robots and task them, the task should take the form of natural language commands and gesture commands. Then, it is essential that the robot understand both kinds of commands and how they correlate with each other. Our project aims to enable robots to understand verbal instructions paired with pointing gestures in navigation problems.

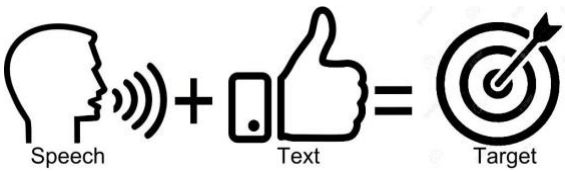

Figure 1: Speech and Text used to generate a target

Previous approaches in navigation problems have focused solely on Natural Language (NL) commands and ignored how gestures can play a part too. Some approaches [1] employ a deep neural network to parse Natural Language references to landmarks, then assess semantic similarities between the referring expression and landmarks in a predefined semantic map of the world. This approach is effective in finding the landmark that's most similar to the user's natural language command and can generalize to new environments outside the training set, but does to take into account gesture commands. Other approaches [2] focus solely on gesture commands and does not take in verbal instructions. Berg et al. developed an interface to enable human to interact with the Skydio R1 drone in Augmented Reality (AR) and use gestures to command to drone to perform different actions. While this is effective in interpreting gesture commands, it does not take into account that humans most often use verbal language to communicate their intensions and would wish to do so with robots.

Our approach in this project aims to combine previous approaches and develop a pipeline that enables simultaneous Natural Language and gesture commands on robots. We will focus on navigation problems, enabling the robot to take in a verbal command and a gesture, and find the destination the user is referring to in the map. Since we focus exclusively on navigation problems, we will limit the gesture input to pointing gestures, which makes the most sense in navigation commands. In terms of verbal language commands, our model can support blurry and vague commands, such as "go to restaurant" or "go to lab" without specifying which restaurant or lab. This is helpful in real-life scenarios where the user can't remember the specific name of a restaurant but knows which direction it's in.

We build our pipeline based on the work of Berg et al. Our model first takes in a user's verbal command, convert it to text, and send the text to Berg et al's deep language model. The language model will compare the similarity of the user's command and landmarks names in the map and output the n-most-likely landmarks. Then, we will compare those top landmarks with the user's pointing gesture and decide which one the user is most likely suggesting.

Figure 2: OpenStreetMap view of Brown University Campus

Our model works well and can help improve the accuracy of the decision when the user's Natural Language command is vague. Moreover, it is easily generalizable and can operate on new environments. Future work on this problem can also easily expand the type of gesture inputs with minimal work.

Related Work

There has been a number of research papers on human-robot collaboration using Natural Language and human gestures.

Berg et al. [1] employed a deep neural network to parse Natural Language references to landmarks, then assess semantic similarities between the referring expression and landmarks in a predefined semantic map of the world. This approach is effective in finding the landmark that's most similar to the user's natural language command and can generalize to new environments outside the training set, but does to take into account gesture commands.

Similarly, Oh et al. [7] also presented a weakly supervised language model that can learn to map Natural English Language to Linear Temporal Logic (LTL) expressions. Their approach can enable robots to understand Natural Language command with sequential constraints. But a similar constraint to their model is that they also don't take into account the role gestures play in giving commands to robots.

Berg et al. [2] presented another approach that focuses solely on gesture commands and does not take in verbal instructions. They developed an interface to enable human interaction with the Skydio R1 drone in Augmented Reality (AR) and use gestures to command to drone to perform different actions. While this is effective in interpreting gesture commands, it does not take into account that humans most often use verbal language to communicate their intensions and would wish to do so with robots.

Technical Approach

Our goal for this project is to enable robots to take in Natural Language and human gesture commands simultaneously and find the most probable location the user is referring to on a map.

Our entire pipeline is built in Unity and is super user-friendly. The user doesn't need to understand how the entire deep network works behind the scene. They just have to picture themselves standing in the middle of the map in Unity and give NL and gesture commands.

First of all, the Unity system will take in a verbal language command from the user, and use the Watson speech to text library to convert the user's verbal language into text.

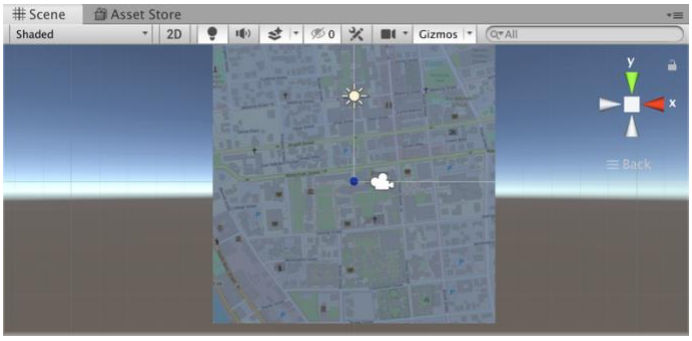

Figure 3: View of Unity project set up with a map of Brown University

Next, we present a networking solution to enable Unity to communicate with the Python terminal. The text-based user command is then sent from Unity to a deep language model in Python made by Berg et al. [1]. The language model will process the inputted command and compare its semantic similarity with all the landmark names in the map. The model will use the fasttext library and transform all English words into multi-dimensional vectors, and use cosine similarity to compare the semantic similarity between different Natural Language words and phrases. In the end, the language model will output a sorted list of all the landmarks names in the map, arranged so that the most semantic-similar landmarks are in the front and the least similar ones are in the back. Then, the list is processed by another Python program to pair each of these landmark names with their latitudes and longitudes. We are using the latitudes and longitude provided by OpenStreetMap, and coordinates of a building are represented by the accurate latitude and longitude of a corner of the building.

Then, the list of landmark names and their latitudes and longitudes are sent back to the Unity project, where the user can choose to display the top few choices on the map.

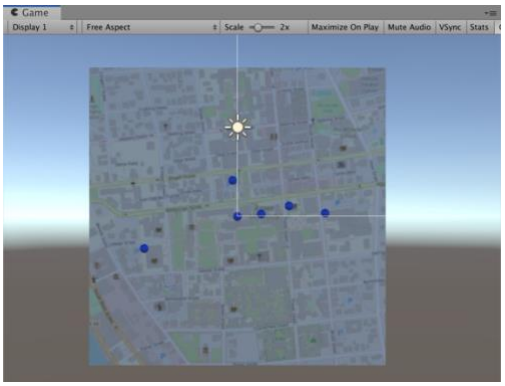

Figure 4: displaying the top5 destinations

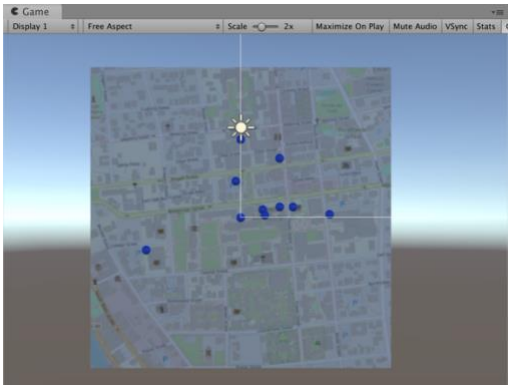

Figure 5: displaying the top10 destinations

Then, the user can input his or her gesture command. Due to restrictions of hardware, we are using a simulated pointing gesture: the user can click on the map in Unity and the model will automatically generate a pointing vector that points from the middle of the map to the click position, to simulate an actual pointing gesture. The Unity project, then, will compare the location of the landmarks displayed on the map to the pointing vector. It will automatically compute the distance of the location of landmarks to the pointing vector's direction, before giving the final result of the closest landmark, and mark it in a red color.

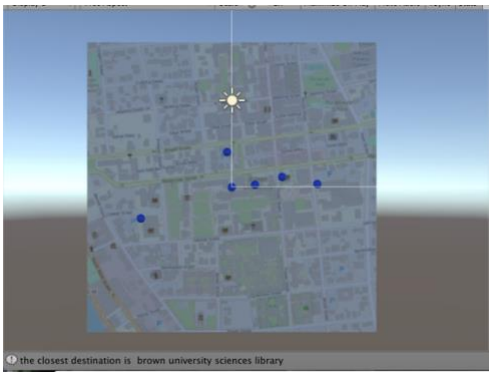

Figure 6: outputting the final destination decided by the model

The user will also be able to see an immersive view of the map in 3D and move around the map using arrow keys, while rotating the camera to see the different possible destinations around them. The user can identify his or her own position in the map by the movement of the big green ball on the left-hand side. This user immersive view enables an easy transition of this project to a version in Virtual Reality (VR).

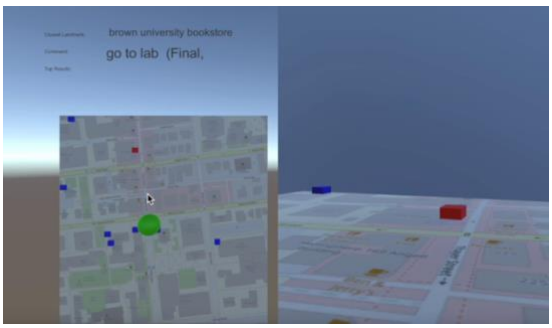

Figure 7: user immersive view of the map (right-hand side)

In addition, we have also been able to use the Vive headsets to recognize many different types of human hand gestures in virtual reality. Due to unanticipated circumstances in COVID-19, the Virtual Reality (VR) has not been connected to the Unity project, but we anticipate it will be a minimum amount of networking work to be done to use gesture inputs in VR instead of using simulated pointing vectors.

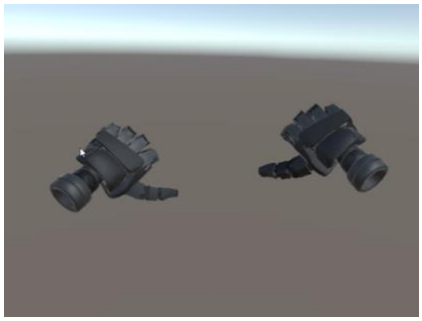

Figure 8 and 9: gesture inputs in Virtual Reality using Vive headsets

Evaluation

Our goal for this project is to enable robots to take in Natural Language and human gesture commands simultaneously and find the most probable location the user is referring to on a map.

This goal is definitely achieved since our Unity project can take in both verbal commands and pointing gesture commands to output a final destination in simulation. We can confirm that the effect of using both language and gesture definitely gives rise to the accuracy of the final decision. For example, if the user gives the command of “go to lab”, the language is vague because the user is not specifying which lab. The deep language model we use will give the result of “medical research lab”, but the user could have well been intending to mean “engineering lab” or “environmental lab”. With the help of a pointing gesture, the model can differentiate between the different “lab”s.

Conclusion

We expand Human-Robot Interaction from solely basing on Natural Language or human gestures to the combination of both. In this project we use a deep language model and simulated pointing gestures in navigation problems, and verify that the combination of verbal and gestural command can return a more accurate result and handle vague languages.

Future work in this area involves improving the deep language model and expanding the types of human gestures. With regard to the language aspect of this project, the weakest link is the running time of the language model. Work to improve the model and make the calculations instant could enable real-time destination-finding. With regard to the gesture aspect, future work could be directed to enable different sorts of gestures, not just pointing gestures, and enable Virtual Reality gesture input using the Vive headsets.

References

[1] Matthew Berg, Deniz Bayazit, Rebecca Mathew, Ariel Rotter-Aboyoun, Ellie Pavlick, and Stefanie Tellex. Grounding Language to Landmarks in Arbitrary Outdoor Environments. In IEEE International Conference on Robotics and Automation (ICRA), 2020.

[2] Matthew Berg and Lilika Makatou, Magic Skydio, May 2019

[3] Michael Walker, Hooman Hedayati, Jennifer Lee, and Daniel Szafir. Communicating robot motion intent with augmented reality. In Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction, HRI '18, pages 316–324, New York, NY, USA, 2018. ACM.

[4] Baichuan Huang, Deniz Bayazit, Daniel Ullman, Nakul Gopalan, and Stefanie Tellex. Flight, Camera, Action! Using Natural Language and Mixed Reality to Control a Drone. In IEEE International Conference on Robotics and Automation (ICRA), 2019.

[5] S. Omidshafiei, A. Agha-Mohammadi, Y. F. Chen, N. K. Ure, S. Liu, B. T. Lopez, R. Surati, J. P. How, and J. Vian. Measurable augmented reality for prototyping cyberphysical systems: A robotics platform to aid the hardware prototyping and performance testing of algorithms. IEEE Control Systems Magazine, 36(6):65–87, Dec 2016.

[6] Natasha Danas, Tim Nelson, Cobi Finkelstein, Shriram Krishnamurthi, Stefanie Tellex. Formal Dialogue Model for Language Grounding Error Recovery, 2019

[7] Yoonseon Oh, Roma Patel, Thao Nguyen, Baichuan Huang, Ellie Pavlick, and Stefanie Tellex. Planning with state abstractions for non-markovian task specifications. arXiv preprint arXiv:1905.12096, 2019.